Density-Based Clustering and Jet Reconstruction

Igor Volobouev

Texas Tech University

igv@highenergy.phys.ttu.edu

Definitions

- I use the term "density-based clustering" to designate a clustering algorithm which employs some kind of a non-parametric probability density estimate. In particular, in this talk I will examine some approaches based on kernel density estimation (KDE) which may turn out to be quite effective for clustering jets.
- In pattern recognition literature, "density-based cluster" is usually defined as a set of spatially connected points at which the density is above a certain cutoff (these are called "core points") combined with all points within certain neighborhood of core points. Points in the cluster with density below the cutoff are called "border points".
- Unlike density-based clustering, "agglomerative clustering" initially assigns each object into its own cluster and then merges clusters if a distance between them is below a certain cutoff (*e.g.*, the k_{t} algorithm).

Kernel Density Estimation (KDE)

- With a set of weighted points $(\mathbf{x}_i, w_i)$, $i = 1, ..., N$, kernel estimate of the weight at point $\mathbf{x}$ is defined as

$$\hat{w}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} w_i K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i)$$

where

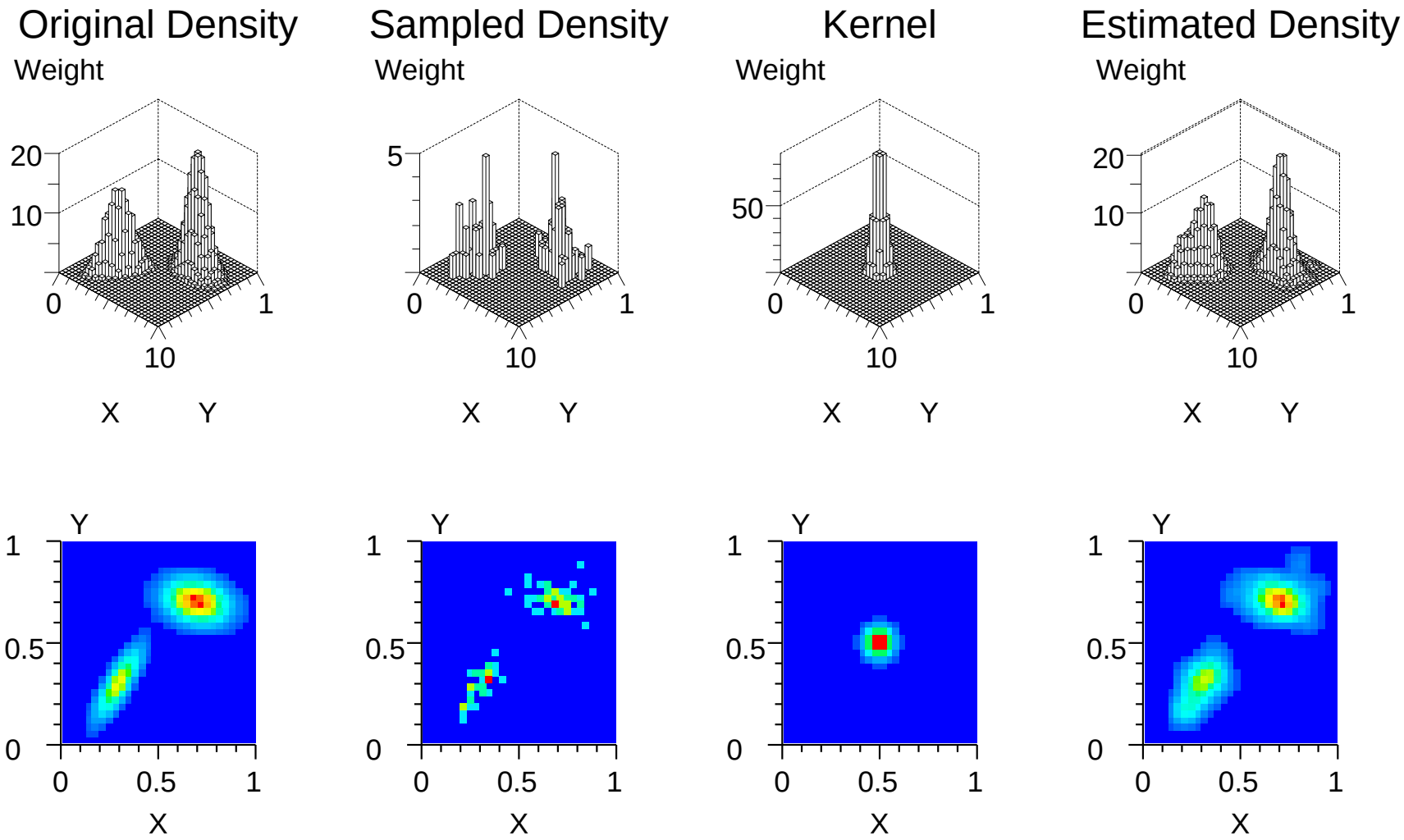
$$K_{\mathbf{H}}(\mathbf{x}) \equiv |\mathbf{H}|^{-1/2} K(\mathbf{H}^{-1/2}\mathbf{x})$$

$\mathbf{H}$ is a symmetric positive definite *bandwidth matrix*, and $K(\mathbf{x})$ is the *kernel* function usually defined in such a way that it is a symmetric probability density with finite support or decaying at infinity faster than any power of $|\mathbf{x}|$. Then $\hat{w}(\mathbf{y})$ is normalized by average weight, and turns into density if all w_i are 1:

$$\int \hat{w}(\mathbf{x}) d\mathbf{x} = \frac{1}{N} \sum_{i=1}^{N} w_i$$

- $\hat{w}(\mathbf{x})$ is a *convolution* of $K_{\mathbf{H}}(\mathbf{x})$ with the empirical density $\frac{1}{N} \sum_{i=1}^{N} w_i \delta(\mathbf{x} - \mathbf{x_i})$

2-Dimensional KDE Example



Mean Shift Clustering

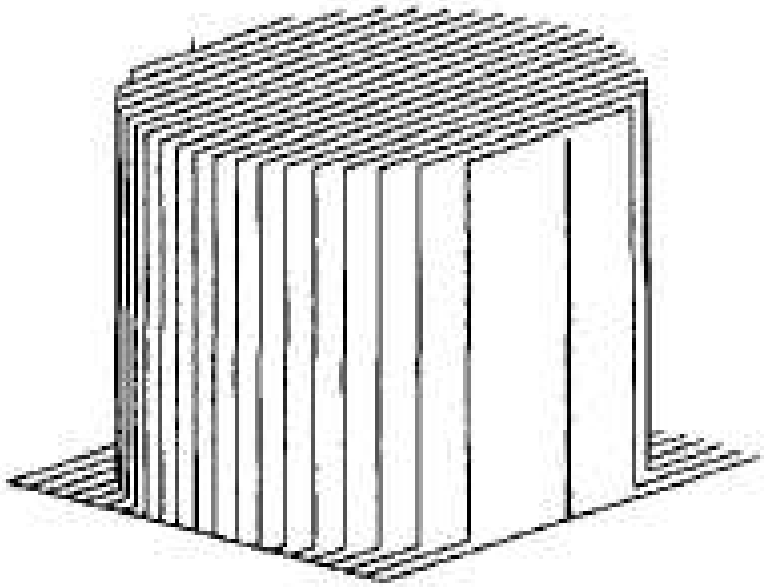
- The cone algorithm we all know is called "Mean Shift" clustering algorithm in the pattern recognition literature. To my knowledge, the first version of mean shift was proposed by Fukunaga and Hosteller in 1975 in a paper which was largely forgotten until 1995.
- Y. Cheng, "Mean Shift, Mode Seeking, and Clustering", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol 17, Aug 1995, p. 790.

  This paper generalized the mean shift algorithm and established a deep connection between mean shift clustering and kernel density estimation.
- For our purposes, the most important result proven in Cheng's paper is that locations of stable cone centers correspond to modes (peaks) of the density estimated with the *"shadow kernel"*. The Epanechnikov kernel:

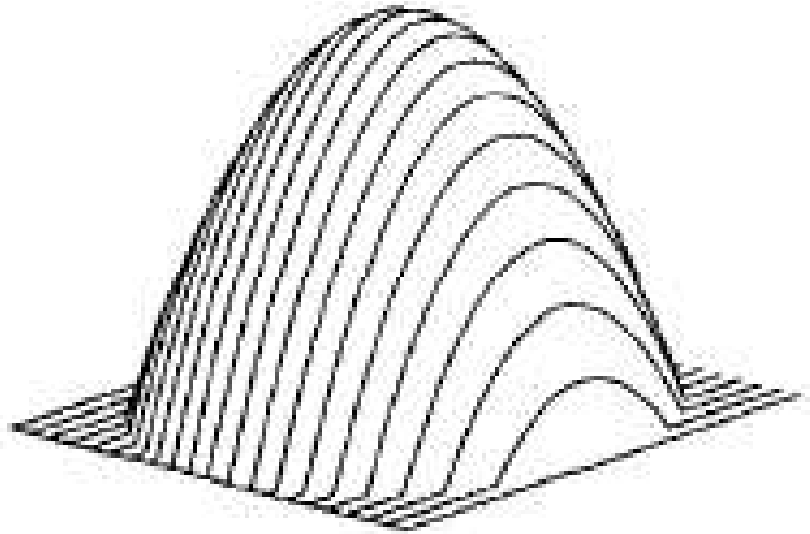
$$K(\mathbf{x}) = \begin{cases} 1 - |\mathbf{x}|^2 & \text{if } |\mathbf{x}| \leq 1 \\ 0 & \text{if } |\mathbf{x}| > 1 \end{cases}$$

  is the shadow of the flat kernel. Please consult Cheng's paper if you want to know how to calculate centroids using kernels and how to construct shadow kernels.

Relevant Kernels (as shown in Cheng's paper)



Flat



Epanechnikov

Fast Seedless Cone Algorithm

The Fast Seedless Cone (FSLC) algorithm in an immediate consequence of the Cheng's result and the fact that KDE is a convolution of the kernel with the empirical density. The algorithm top-level steps are:

1. Discretize the calorimeter signals (or MC particles) on a regular grid in the η-ϕ space suitable for Discrete Fast Fourier Transform (DFFT).
2. Convolute discretized signals with the Epanechnikov kernel using $diag(R^2)$ as the bandwidth matrix. Do it via DFFT.
3. Find modes of the obtained density estimate. These are the locations of the stable cone centers.
4. (A simplistic approach) Find the energies inside the cones by performing another convolution, this time with the flat kernel.

Notes on FSLC

- FSLC is just fast SLC, so it has all theoretical properties of SLC (infrared and collinear safety, underlying event can be easily subtracted, *etc.*)
- The algorithm complexity is $N \log(N)$ where N is the number of nodes in the discretization grid. In practice, N will be comparable to the number of towers in a detector calorimeter.
- Fourier images of the kernels can be built only once and then can be reused for all events.
- The precision of jet direction determination in step 3 can be improved by locally fitting the density near the modes with a quadratic surface.
- In the form shown on the previous page, step 4 ignores cone overlaps and implements the simplest possible recombination scheme in which the energy of the whole cluster is assigned to the direction of the cluster centroid. Better recombination and overlap resolution schemes can be introduced by performing a lookup of all towers within radius R from the cone centers in the η-ϕ space and then combining these towers in some other ways (*e.g.*, adding 4-momenta).

Improving FSLC

- Within the paradigm of reconstructing jet direction via KDE it is reasonable to pose the following question: is the Epanechnikov kernel optimal for determining the jet direction? Or, equivalently, can we do something better than calculating the cone centroid?
- I think that, in the presence of any noise or pile-up, a kernel which looks like an average lateral jet profile will work better than the Epanechnikov kernel. In fact, if something definite is known about the noise and pile-up characteristics, one can apply an optimal Wiener filter in the Fourier space (the precise definition of the term can be found, *e.g.*, in the "Numerical Recipes" book). This approach also solves the pesky problem of determining optimal calorimeter energy thresholds — there should be no thresholds since the algorithm always works on the full η-ϕ grid anyway and the noise is filtered out efficiently.

KDE-Based Jet Clustering

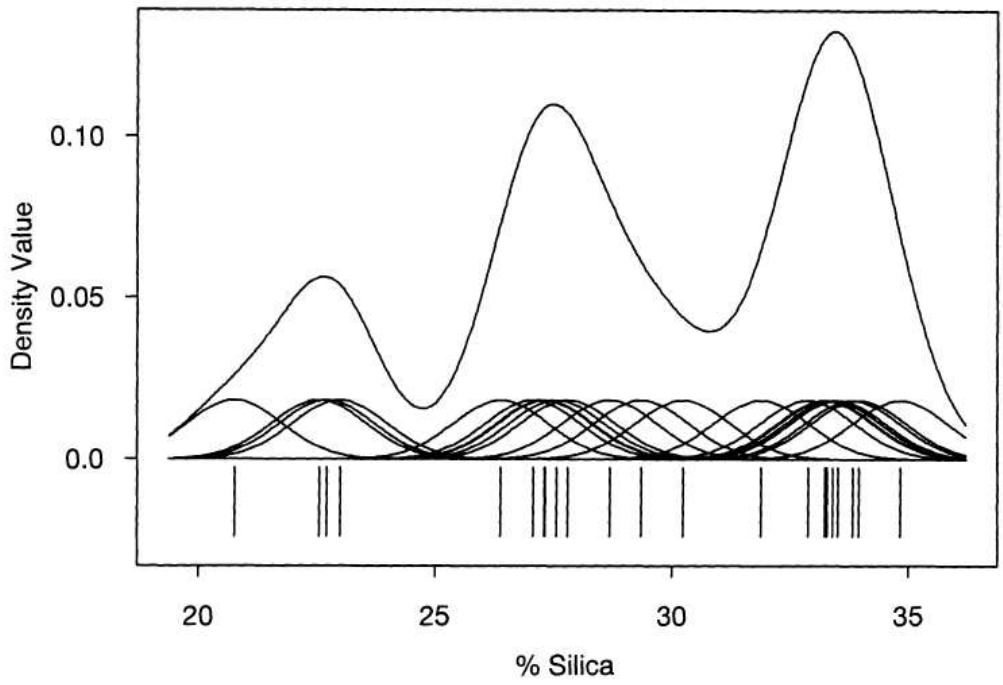
- How KDE-based jet clustering can be extended beyond FSLC?
- By using arbitrary jet shapes. Once the density surface is created with the jet direction kernel, it becomes easy to create density-based clusters and extend the algorithm to arbitrary jet shapes. This flexibility, however, comes at a price: one has to introduce another parameter into the algorithm which is the density cutoff for the cluster core. The optimal value for this parameter will be affected by quantities which are not directly relevant to underlying physics (noise, pile-up). Calibration will become more complicated since each cluster will have its own area and its own peak/cutoff ratio.
- By calculating correlations for nearby jets. When two jets overlap, jet profile kernels placed at the jet centers provide natural probability measure for the in-between energy depositions to belong to one jet or the other. Dividing these depositions between the jets with appropriate weights will, on average, result in a better jet energy determination. One should also calculate the correlation coefficient for jet energies (and, perhaps, directions) due to such depositions which can later be used to improve dijet invariant mass resolution.
- With the multiscale approach.

The Multiscale Approach

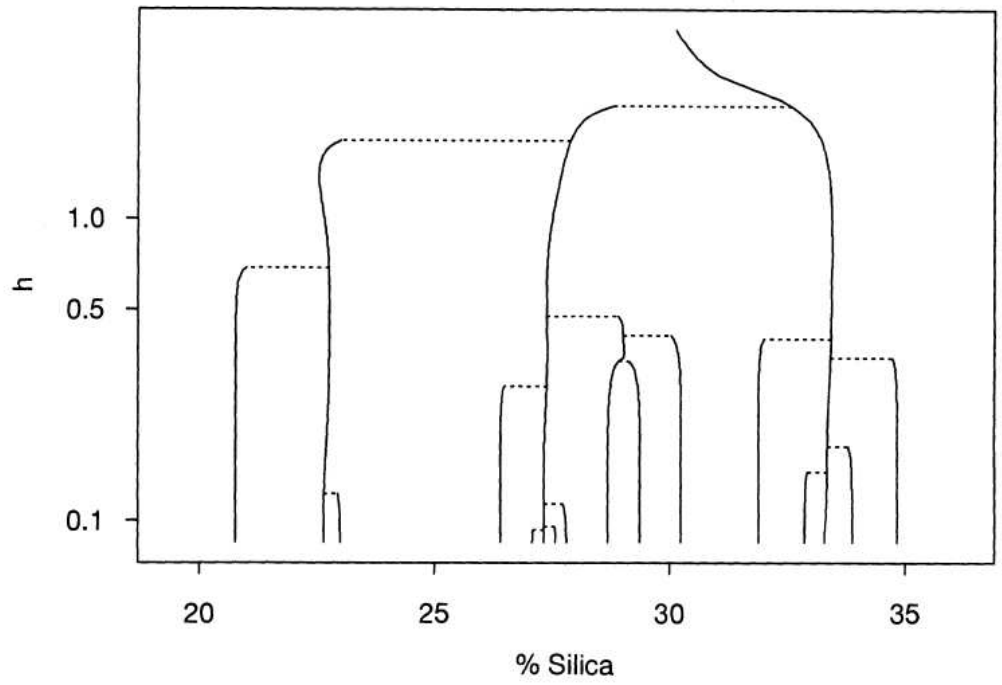
- The multiscale/multiresolution approaches to image recognition problems became very popular with the invention of wavelets. Typing word "multiresolution" in Google returns about two million hits. The idea is to transform the image and see what happens to the image when high frequency/high detail components are first removed and then gradually added back (also google "image pyramid"). It turns out that this type of analysis can be very useful for image compression and denoising.
- A similar approach can be applied to jet clustering. In particular, the bandwidth matrix used for jet direction determination can be varied, and the resulting jet configurations examined as a function of this matrix. Essentially, we are looking at jets in HEP events using all possible cone sizes simultaneously.
- The results of such an analysis can be arranged in a structure known in the statistical literature as "mode tree" (M.C. Minnotte and D.W. Scott, "The Mode Tree: a Tool for Visualization of Nonparametric Density Features", J. Comp. Graph. Statist., Vol 2, 1993, p. 51). Mode tree is, basically, a collection of coordinates of density peaks constructed as a function of bandwidth. It usually ends up looking very similar to a typical hierarchical clustering dendrogram.

Mode Tree (Minnotte and Scott)

KDE with $h = 1.0$



Mode Tree



Why Multiscale?

When a complete mode tree is built for an event and associated jet energies are reconstructed, a variety of interesting and, perhaps, non-traditional approaches to data analysis become accessible. For example, one can start asking questions like this:

- What is the optimal cone size for this particular jet which maximizes S/N ratio?
- How stable is the energy of a jet with respect to change in the cone size? Or how stable is a quantity like dijet mass? One can select events in which the quantity of interest is sufficiently stable and thus reduce the systematic error.
- If one wants to treat this particular event as an event which has exactly four jets, what are the energies and directions of these jets?
- If this particular event has four jets between cone radii R_1 and R_2, how probable are the R_1 and R_2 values compared to the MC simulation of the process under study?

The jets now have a whole new dimension which can be profitably explored. Also, mode trees trivialize jet substructure and fragmentation analyses in the laboratory frame.

Practical Considerations for Multiscale Analysis

- The usable cone radii are obviously limited from below by the calorimeter granularity and from above by π. In practice the usable range is going to be smaller, something like 0.1 to 1.5 comes to mind, where the lower limit should be chosen taking into account the CPU speed limitations. The set of cone sizes used with each event should be adaptive and should depend on the event geometry: if one finds that the number of jets is the same and their directions are very similar for some cone radius R_1 and some other radius R_2, it is not going to be very interesting to examine the radii in between.
- If the CPU speed becomes a problem, one can build dedicated hardware for running this type of analysis. Commercial DFFT processors are available both as ASICs and as FPGA cores. Free codes are available on the web for running DFFT on GPUs in video cards.
- Calibrations are certainly more complex here than in the case of fixed cone size. Still, I think it should be easier to calibrate jet energy as a function of cone size than to calibrate an algorithm which reconstructs arbitrary jet shapes.

Summary

- Density-based clustering is a different way to perform the jet clustering task, and it deserves a good look. It can be used to improve the speed of existing algorithms (FSLC), to introduce improvements (Wiener filtering), or to try new concepts (multiscale analysis).
- Some of the ideas discussed in this talk are probably too far-fetched, too difficult to implement, or too unphysical. I'll appreciate any comments from the experts.
- There is no implementation yet. Please let me know if you are interested in collaborating and bringing some of these ideas to life.